University of Saskatchewan Department of Computer Science

CMPT 215.3 MIDTERM EXAMINATION

November 6th, 2001

Total Marks: 50

CLOSED BOOK and CLOSED NOTES

NO CALCULATOR

Time: 75 minutes

Instructions

Read each question carefully and write your answer legibly on the examination paper. No other paper will be accepted. You may use the backs of pages for rough work but all final answers must be in the spaces provided. The marks for each question are as indicated. Allocate your time accordingly.

Ensure that your name AND student number are clearly written on the examination paper and that your name is on every page.

Note: a reference table of MIPS instructions is provided at the end of the examination paper.

Question	Marks
1 (5 marks)	4
2 (12 marks)	9
3 (18 marks)	io
4 (15 marks)	7.
Total	(30/50)

Name:			
	- , -	1 N N N N N N N N N N N N N N N N N N N	_
Student Number:	930772		

- 1. General (5 marks) Give the technical term that best fits each of the following descriptions or definitions.
 - (a) A special purpose register (on a MIPS system, not directly accessible to the machine language programmer) that stores the address of the next instruction to be executed.

Acromolator

X

- (b) An algorithm for multiplication of signed integers in which a run of consecutive "1"s in the multiplier is handled with just two arithmetic operations (plus some shifts): a subtraction of the multiplicand at the beginning of the run, and an addition of the multiplicand just after the end of the run. Booth's algorithm
- (c) A program chosen to serve as the basis of performance comparison between computer Berchmark
- (d) A numbering system with 16 digits, which are represented by the symbols 0-9, and A,B,C,D,E, and F. Hexaccion
- (e) A law stating that $\overline{A} + \overline{B} = \overline{A \cdot B}$. De Morgans law

2. Computer Performance (12 marks in total)

(a) (2 marks) Suppose that the MIPS rating for a particular program is 450, and the clock rate is 300 MHz. What is the CPI?

C4 14	NT	930	121
Student	Number:	1 20	12

- (b) (6 marks) In each of the following parts, state which ones of the three factors determining CPU execution time (number of machine language instructions executed, clock cycle time, CPI) may change, when the indicated system change is made (and all else is held constant).
 - 7/(i) a clock with higher frequency is used number of marking larguage institutions for routed, clock cate time.
 - a new machine language instruction is implemented for an operation that was previously done in software (i.e., with a sequence of simpler instructions); this is done without impacting the implementation of the existing instructions or their execution times
- (c) (2 marks) Give a formula for the geometric mean of three numbers T1, T2, and T3, and state one motivation for using the geometric mean (rather than the arithmetic mean) when computing a single number summarizing system performance.

(d) (2 marks) Consider a system with two classes of instructions. Class A instructions have a CPI of 2, and class B instructions have a CPI of 8. Suppose that it is possible to reduce the CPI of class B instructions to 5, but at the cost of an increase in the clock cycle time. If 1/3 of the instructions in a particular program are of type A, and 2/3 are of type B, what would be the maximum factor by which the clock cycle time could increase, without increasing the program's execution time?

A 2 2 D 8 s

(1415 - ().x/s) . (8 x/s) Simplify = 7/3 . 142 : 143 = 6 (N. (2x/3) + (5x7) = 43 + 141 - 179 = 4

The clack cycle time could increase about 1.5 times whether the execution time would be tilthrent.

930177

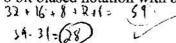
3. Arithmetic (18 marks in total)

(a) (8 marks) Give the base 10 number that is represented by 111011, assuming each of the following representations:

25: 32

(i) 6 bit 2's complement $(-1 \times 2^5) + (1 \times 2^4) + (1 \times 2^3) + (1 \times 2^4) + (1$

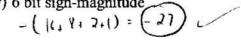
(ii) 6 bit biased notation with bias of 31



(iii) 6 bit 1's complement 000100 =

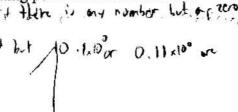


(iv) 6 bit sign-magnitude



(b) (2 marks) How can one tell that a number in the IEEE 754 floating point standard format is a denormalized number? It is a denormalized number lift them is my number but of zero

in front of the decimal point. Intall's assembled but 10 - 120 or 0.11210° or



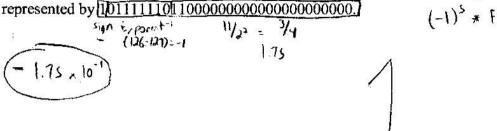
(c) (2 marks) Consider the addition of 4 bit values $a_3a_2a_1a_0$ and $b_3b_2b_1b_0$. Give a logic function for the "carry-in" to the most significant bit position, in terms of quantities $g_i = a_i \cdot b_i$, $p_i = a_i + b_i$, and the "carry-in" to the least significant bit position CarryIn₀.

(Kiry-in: 91 + 92 . 93 . 54

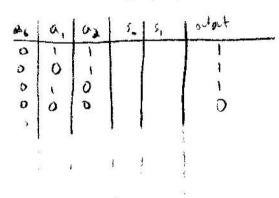
(acry-in =







(e) (4 marks) Give a truth table for a "3 data input, 1 output" multiplexor with data inputs a_0 , a_1 , and a_2 , and select inputs a_0 and a_1 . Suppose that a_0 always has the value 0 (so you don't need to show any rows in your truth table for $a_0=1$). Clearly state any assumptions you need to make. Then, using your truth table, derive a logic equation in sum-of-products form. (You don't need to simplify it.)





Name:	
vaine	

the operation is complete.

Page 6

Student Number: 030772

7 4. Machine and Assembly Language (15 marks in total)

D(a) (4 marks) What functions does a linker perform, and how does it carry out these functions?

A linker is an instruction that links two parts of a program so you improve to a new part of the program and then come back. It is also used in jump instructions and loops. It saves the address of the target and returns to the target and

(b) (2 marks) Consider a proposed new instruction "memmov". For example, "memmov A, B", where A and B are symbolic names, would copy the contents of the memory location with address corresponding to B, to the memory location with address corresponding to A. What difficulty would arise if one tried to add this new instruction to MIPS machine language?

The world have to add a new field to MIPS that could harde the Memory Address.

Student Number: + Nikoly faymen 1 930772

(c) (5 marks) Consider the following code fragment.

.align 2

.word 2, 4, 6, 8, 10

.word 1, 3, 5, 7, 9

.text

main: la \$t0, A la \$t1, B

> move \$t2, \$t1 # \$t2 is the end address of array A

loop: lw \$t7, 0(\$t0) sw \$t7, 0(\$t1)

addi \$t0, \$t0, 4 addi \$t1, \$t1, 4 bne \$t0, \$t2, loop

(i) Following execution of the above code, what are the contents of the 10 words in the data segment? A: >, 4, 6, 8, 10

1:), 4, 6, 1, 10

(ii) How would the result change, if at all, if the loop was changed to the following?

loop: 1b \$t7, 0(\$t0)

sb \$t7, 0(\$t1)

addi \$t0, \$t0, 1

addi \$t1, \$t1, 1 bne \$t0, \$t2, loop

The realt would not change

(iii) If the running time of the original loop is T, what is the running time of the loop in question (ii)?



- (d) (4 marks) Procedure calls in MIPS should follow certain conventions. Answer the following two questions using the procedure call conventions discussed in class and in the text.
- (i) Consider a procedure foo with five integer arguments. Supposing that a routine wants to call foo with the contents of \$s0, \$s1 and \$s2 for the first three parameters, and the values 3 and 7 for the 4th and 5th parameters, write a sequence of instructions that makes the call and passes the parameters according to the MIPS conventions.

(ii) Suppose that the procedure foo makes a call to another procedure, and then uses the result from that procedure call to compute its own return result. When computing its own return result, it uses registers \$s0 and \$t0 (changing the values in these registers). Write a sequence of instructions for use at the beginning of foo, that saves registers to the stack as is necessary.

subi tsp, tsp, 8 hu tso, 4(tsp) hu tto, 0(tsp)

MIPS machine language

		Example						
Name	Format	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	Comments
add	R	0	2	3	1	0	32	add \$1,\$2,\$3
sub	·R	0	2	3	1	0	34	sub \$1,\$2,\$3
addi	1, 12	8	2	1		100		addi \$1.\$2.100
addu	R	0	2	3	1	0	33	addu \$1,\$2,\$3
subu	R	0	2	3	1	0	35	subu \$1,\$2,\$3
addiu	1	9	• 2	1	g 12	100	·,	addiu \$1,\$2,100
mfc0	R	1.6	0	1 .	14		0	mfc0 \$1.\$epc
mult	R	0	2	. 3	0	0	- 24	mult \$2,\$3
multu	R	0	. 2	3	0.	0 '-	⁻⁻ 25	multu \$2,\$3
div	, R :	0	2 :	- 3 -	0	. 0	 26	div \$2,\$3
divu	R	0	- 2	3.	0	0	27	divu \$2,\$3
mfhi	R	0	0	0	1	0;	7ª 16 .	mfhi \$1
mflo	R	0	0 .	. 0	1	0	18	mflo \$1
and	R	0	2	3	1	0	36	and \$1,\$2,\$3
or	R	0	2	. 3	1	0	*:.37	or \$1,\$2,\$3
andi	i	12	2	1	1.	100		andi \$1,\$2,100
ori	1	13	2	1		1.00 -	7.50	ori \$1,\$2,100
s11 ·	R	. 0	0	. 2 :*	1		O	\$1,\$2,10
srl	R	0	.0	2	- 1	10	- 2	srl \$1,\$2,10
1w		35	2	. 1		100 🔩	- g/u	lw \$1,100(\$2)
SW W2	. 1	43 .	2	1.			S. W. s.	sw \$1,100(\$2)
lui	- 1	15	0	1		100	ow,	lui \$1,100
beq	ī	4	1	2		25		beq \$1,\$2,100
bne ·		5	1 .	2		25	era -	bne \$1,\$2,100
slt .	R	-0	2	.3	1	0	. 42	slt \$1,\$2,\$3
slti	. , 1	10	2	1	7 7 2	100		slti \$1.\$2.100
sìtu	R ·	0	·· 2	3	1	- 0 😁	€ 43	sltu \$1,\$2,\$3
sltiu	1	11	2 .	1	* *	100	, h	sltiu \$1,\$2,100
j	J	2		-	2500		4"	j 10000
jr	R	0	31	0	0	0 . 1	: 8	jr \$31
jal	J	3			2500	0.0		jal 10000

MIPS instruction formats

Name			Comments				
Field size	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	All MIPS instructions 32 bits
R-format	ор	rs	rt	rd ~	shamt	funct	Arithmetic instruction format
I-format	ор	rs	rt	address/immediate			Transfer, branch, imm. format
J-format	ор	1,000	1	target addre	Jump Instruction format		

Main MIPS machine language. Formats and examples are shown, with values in each field: op and funct fields form the opcode (each 6 bits), rs field gives a source register (5 bits), rt is also normally a source register (5 bits), rd is the destination register (5 bits), and shamt supplies the shift amount (5 bits). The field values are all in decimal. Floating-point machine language instructions are shown in Figure 4.47 on page 291. Appendix A gives the full MIPS machine language.

MIPS operands

Name	Example	Comments
32 registers		Fast locations for data. In MIPS, data must be in registers to perform arithmetic. MIPS register \$zero always equals 0. Register \$at is reserved for the assemble to handle large constants. Hi and Lo contain the results of multiply and divide.
20 27	Memory[4], ,	Accessed only by data transfer instructions. MIPS uses byte addresses, so sequential words differ by 4. Memory holds data structures, such as arrays, and spilled registers, such as those saved on procedure calls.

MIPS assembly language

Category	Instruction		Example	Meaning	Comments
	add	add	\$\$1,\$\$2,\$\$3	\$51 = \$52 + \$53	Three operands; overflow detected
	subtract	sub	\$\$1,\$\$2,\$\$3	\$s1 = \$s2 - \$s3	Three operands; overflow detected
	add immediate	addi	\$\$1,\$\$2,100	\$s1 = \$s2 + 100	+ constant; overflow detected
	add unsigned	addu	\$\$1.\$\$2.\$\$3	\$\$1 = \$\$2 + \$\$3	Three operands; overflow undetected
	subtract unsigned	subu	\$\$1.\$\$2.\$\$3	\$51 = \$52 - \$53 /	Three operands; overflow undetected
	add immediate unsigned	addiu	\$\$1,\$\$2,100	\$s1 = \$s2 + \$s3	+ constant; overflow undetected
Arithmetic	move from coprocessor register	mfc0	\$s1.\$epc	\$s1 = \$epc	Used to copy Exception PC plus other special registers
	multiply	mult	\$\$2.\$\$3	Hi, Lo = \$52 x \$53	64-bit signed product in Hi, Lo
	multiply unsigned	multu	\$52,\$53	Hi, Lo = \$52 x \$53	64-bit unsigned product in HI, Lo
ĸ	divide	div	\$52.\$53	Lo = \$52 / \$53, Hi = \$52 mod \$53	Lo = quotient, HI = remainder
	divide unsigned .	divu	\$52,\$53	Lo = \$s2 / \$s3 Hi = \$s2 mod \$s3	Unsigned quotient and remainder
	move from HI	mfhi	\$ s1	\$s1 = Hi	Used to get copy of Hi
	move from Lo	mflo	\$s1	\$51 = Lo -	Used to get copy of Lo
	and	and	\$\$1,\$\$2,\$\$3	\$\$1 = \$\$2 & \$\$3	Three reg. operands; logical AND
	or	or	\$\$1,\$\$2,\$\$3	\$51 = \$521\$53	Three reg. operands; logical OR
Lagiani	and immediate	andi	\$s1.\$s2,100	\$s1 = \$s2 & 100	Logical AND reg. constant
Logical	or immediate	ori	\$\$1.\$\$2,100	\$51 = \$521100	Logical OR reg, constant
	shift left logical	511	\$\$1.\$\$2,10	\$\$1 = \$\$2 << 10	Shift left by constant
	shift right logical	srl	\$s1.\$s2.10	\$s1 = \$s2 >> 10	Shift right by constant
	load word	1w	\$\$1,100(\$\$2)	\$s1 = Memory(\$s2+100)	Word from memory to register
Data	store word	SW-	\$s1.100(\$s2)	Memory(\$52 + 100] = \$51	Word from register to memory
14-000-0	load byte unsigned	1 bu	\$\$1,100(\$\$2)	\$s1 = Memory[\$s2 + 100]	Byte from memory to register
transfer	store byte	sb	\$51,100(\$52)	Memory[\$52 + 100] = \$51	Byte from register to memory
	load upper immediate	lui	\$s1,100	\$s1 = 100 * 2 ¹⁶	Loads constant in upper 16 bits
	branch on equal	beq	\$\$1.\$\$2,25	if (\$s1 == \$s2) go to	Equal test; PC-relative branch
	branch on not equal	bne	\$\$1,\$\$2,25	If (\$51 l= \$52) go to PC + 4 + 100	Not equal test; PC-relative
Conditional	set on less than	slt	\$s1,\$s2,\$s3	If (\$52 < \$53) \$51 = 1; else \$51 = 0	Compare less than; two's complement
branch	set less than immediate	slti	\$s1,\$s2,100	if (\$52 < 100) \$51 = 1; else \$51=0	Compare < constant; two's complement
AT.:	set less than unsigned	sltu	\$\$1.\$\$2,\$\$3	if (\$s2 < \$s3) \$s1 = 1; else \$s1=0	Compare less than; natural numbers
	set less than immediate unsigned	sltiu	\$s1,\$s2,100	if (\$52 < 100) \$51 = 1; etse \$51 = 0	Compare < constant; natural numbers
Unconditional	jump	j	2500	go to 10000	Jump to target address
ACC 16.57-604	jump register	jr	\$ra	go to \$ra	For switch, procedure return
jump	jump and link	ja1	2500	\$ra = PC + 4; go to 10000	For procedure call

Main MIPS assembly language instruction set. The floating-point instructions are shown in Figure 4.47 on page 291. Appendix A gives the full MIPS assembly language instruction set.